SPOS Assignment No.4(Page Translator)

1. FIFO (First In First Out)

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class FIFO
{
    public static void main(String[] args) throws IOException
    {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int frames, pointer = 0, hit = 0, fault = 0, ref_len;
        int[] reference, buffer;
        int[][] mem_layout;

        System.out.println("Please enter the number of frames:");
        frames = Integer.parseInt(br.readLine());

        System.out.println("Please enter the length of the reference string:");
        ref_len = Integer.parseInt(br.readLine());

        reference = new int[ref_len];
        mem_layout = new int[ref_len][frames];
        buffer = new int[frames];

        for (int j = 0; j < frames; j++)
            buffer[j] = -1;

        System.out.println("Please enter the reference string:");
        for (int i = 0; i < ref_len; i++)
        {
            reference[i] = Integer.parseInt(br.readLine());
        }
        System.out.println();

        for (int i = 0; i < ref_len; i++)
        {
            int search = -1;
            for (int j = 0; j < frames; j++)
            {
                if (buffer[j] == reference[i])
                {
                    search = j;
                    hit++;
                    break;
                }
            }

            if (search == -1)
             {
                buffer[pointer] = reference[i];
                fault++;
                pointer = (pointer + 1) % frames;
             }

             for (int j = 0; j < frames; j++)
                    mem_layout[i][j] = buffer[j];
          }

          for (int i = 0; i < frames; i++)
          {
                    for (int j = 0; j < ref_len; j++)
                    {
                          System.out.printf("%3d ", mem_layout[j][i]);
                    }
                    System.out.println();
          }
          System.out.println("The number of Hits: " + hit);
          System.out.println("Hit Ratio: " + (float) hit / ref_len);
          System.out.println("The number of Faults: " + fault);
      }
}
```

2. LRU (Least Recently Used)

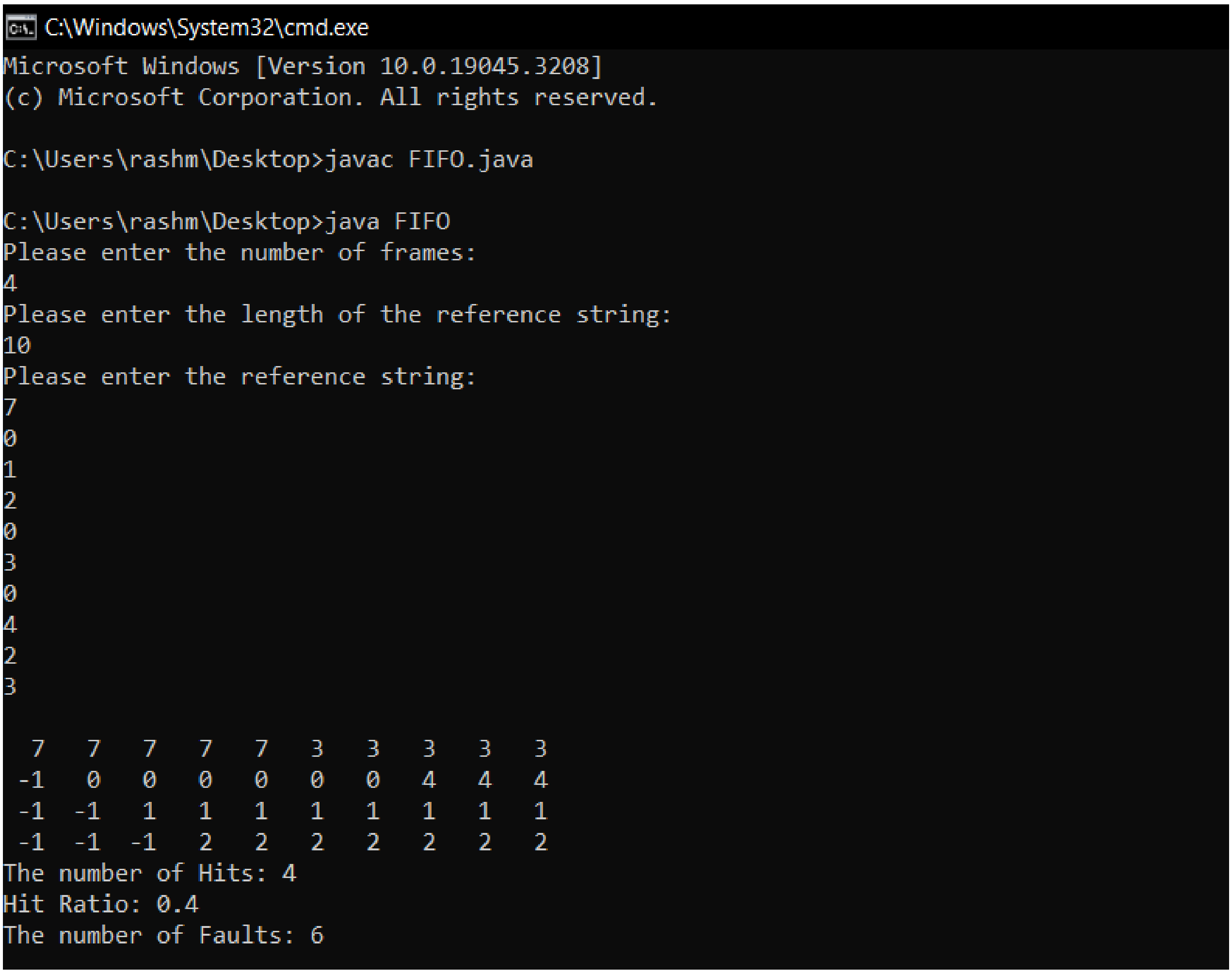
```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.LinkedList;

public class LRU
{

    public static void main(String[] args) throws IOException
    {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int frames, pointer = 0, hit = 0, fault = 0, ref_len;
        boolean isFull = false;
        int[] buffer;

        System.out.println("Please enter the number of frames:");
        frames = Integer.parseInt(br.readLine());

        System.out.println("Please enter the length of the reference string:");
        ref_len = Integer.parseInt(br.readLine());

        int[] reference = new int[ref_len];
        int[][] mem_layout = new int[ref_len][frames];
        buffer = new int[frames];

        for (int j = 0; j < frames; j++)
            buffer[j] = -1;

        System.out.println("Please enter the reference string:");
        for (int i = 0; i < ref_len; i++)
        {
            reference[i] = Integer.parseInt(br.readLine());
        }
        System.out.println();
        LinkedList<Integer> stack = new LinkedList<>();

        for (int i = 0; i < ref_len; i++)
        {
            if (stack.contains(reference[i]))
            {
                stack.remove(stack.indexOf(reference[i]));
            }
            stack.add(reference[i]);
            int search = -1;
            for (int j = 0; j < frames; j++)
            {
                if (buffer[j] == reference[i])
                {
                    search = j;
                    hit++;
                    break;
                }
```

```
              }
              if (search == -1)
              {
                    if (isFull)
                    {
                          int min_loc = ref_len;
                          for (int j = 0; j < frames; j++)
                          {
                                 if (stack.contains(buffer[j]))
                                 {
                                        int temp = stack.indexOf(buffer[j]);
                                        if (temp < min_loc)
                                        {
                                              min_loc = temp;
                                              pointer = j;
                                        }
                                 }
                          }
                    }
                    buffer[pointer] = reference[i];
                    fault++;
                    pointer++;
                    if (pointer == frames)
                    {
                          pointer = 0;
                          isFull = true;
                    }
              }
              for (int j = 0; j < frames; j++)
                    mem_layout[i][j] = buffer[j];
        }

        for (int i = 0; i < frames; i++)
             {
                    for (int j = 0; j < ref_len; j++)
                    {
                          System.out.printf("%3d ", mem_layout[j][i]);
                    }
                    System.out.println();
             }
             System.out.println("The number of Hits: " + hit);
        System.out.println("Hit Ratio: " + (float) hit / ref_len);
        System.out.println("The number of faults: " + fault);
    }
}
```

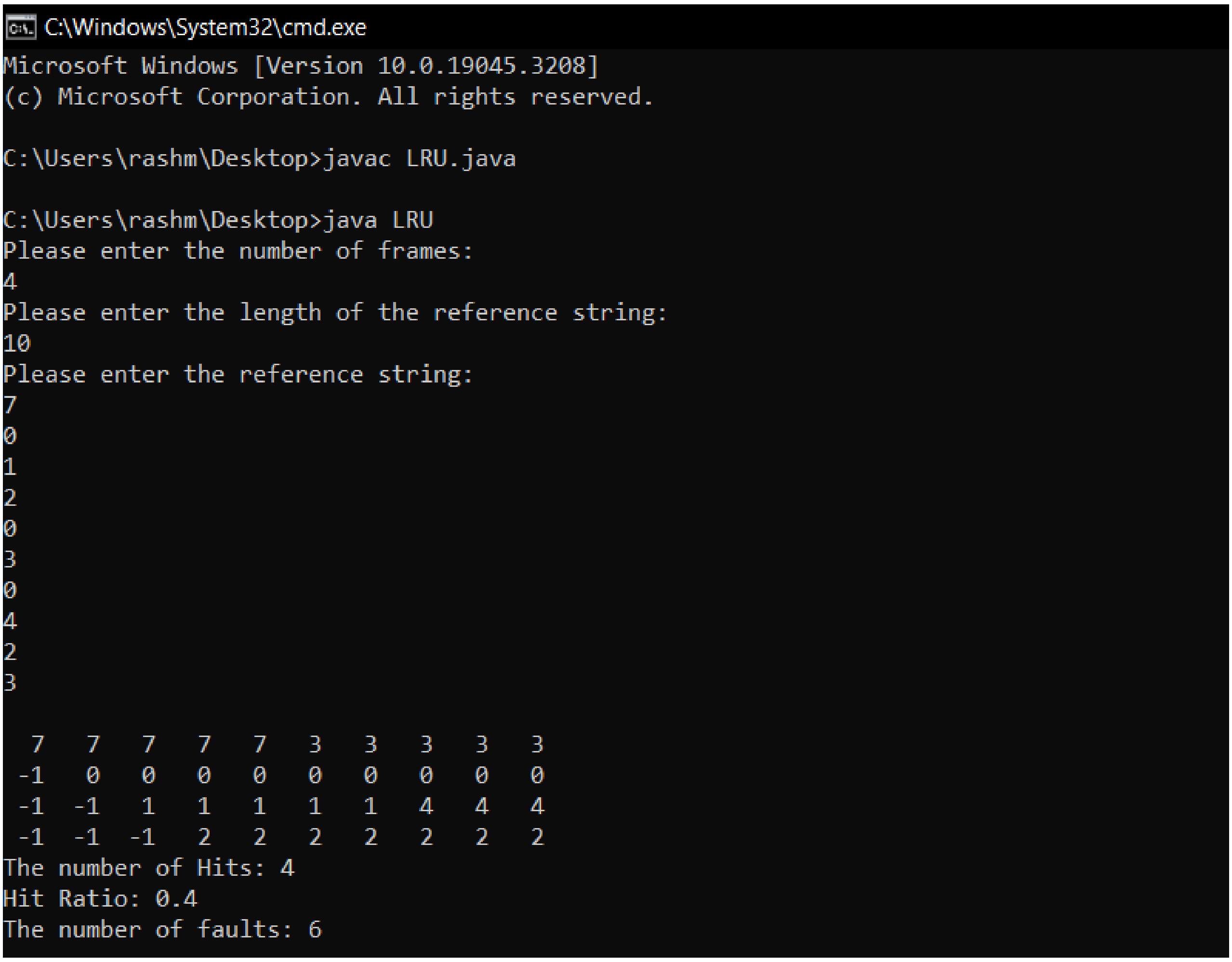
3. Optimal Algorithm

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class Optimal
{
    public static void main(String[] args) throws IOException
    {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int frames, pointer = 0, hit = 0, fault = 0, ref_len;
        boolean isFull = false;
        int[] buffer;
        int[] references;
        int[][] mem_layout;

        System.out.println("Please enter the number of frames:");
        frames = Integer.parseInt(br.readLine());

        System.out.println("Please enter the length of the References string:");
        ref_len = Integer.parseInt(br.readLine());

        references = new int[ref_len];
        mem_layout = new int[ref_len][frames];
        buffer = new int[frames];
        for (int j = 0; j < frames; j++)
        {
            buffer[j] = -1;
        }

        System.out.println("Please enter the references string:");
        for (int i = 0; i < ref_len; i++)
        {
            references[i] = Integer.parseInt(br.readLine());
        }

        for (int i = 0; i < ref_len; i++)
        {
            int search = -1;
            for (int j = 0; j < frames; j++)
            {
                if (buffer[j] == references[i])
                {
                    search = j;
                    hit++;
                    break;
                }
            }
            if (search == -1)
             {
                    if (isFull)
                    {
```

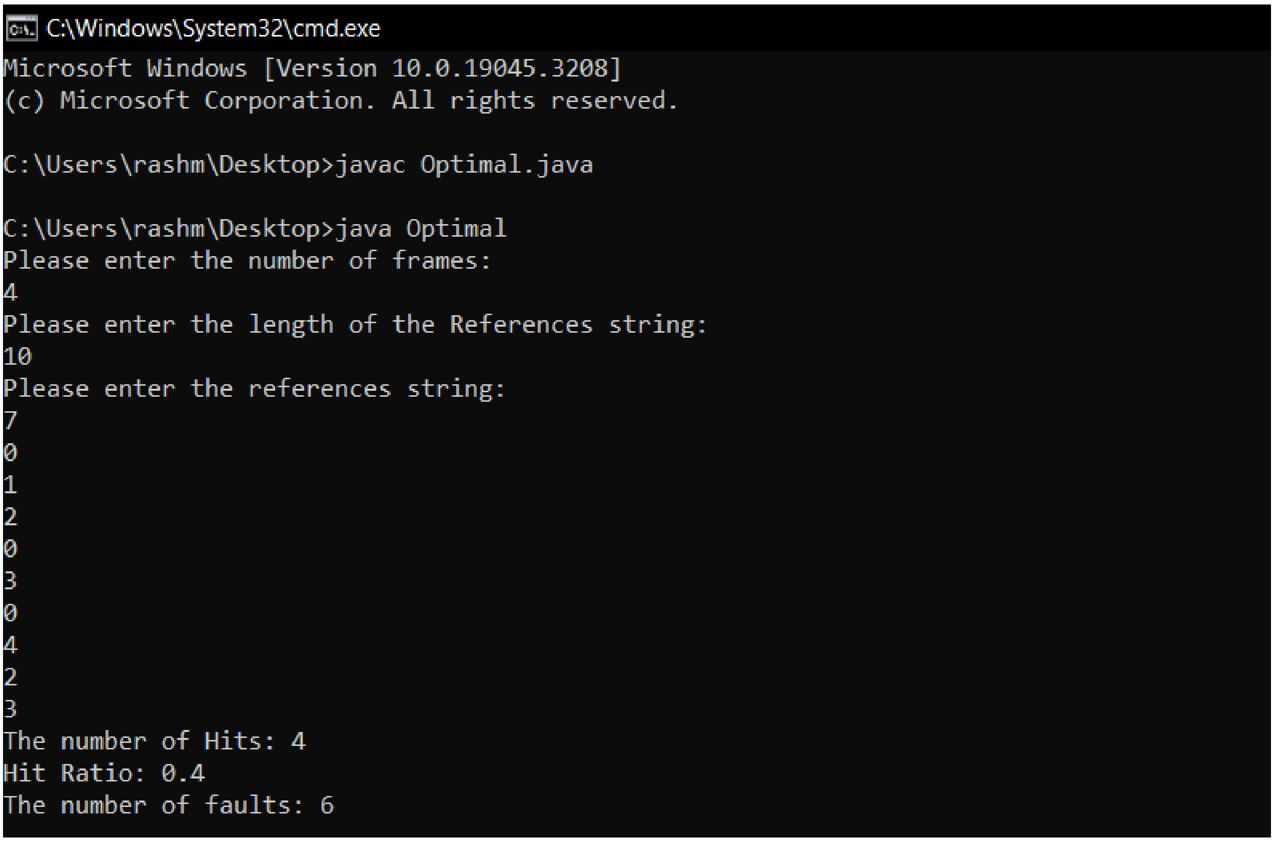
```
                    int[] index = new int[frames];
                    for (int j = 0; j < frames; j++)
                    {
                        index[j] = -1;
                    }

                    for (int j = i + 1; j < ref_len; j++)
                    {
                        for (int k = 0; k < frames; k++)
                        {
                            if (references[j] == buffer[k] && index[k] == -1)
                            {
                                index[k] = j;
                            }
                        }
                    }

                    int max = index[0];
                    pointer = 0;
                    for (int j = 1; j < frames; j++)
                    {
                        if (index[j] == -1)
                        {
                            pointer = j;
                            break;
                        }
                        if (index[j] > max)
                        {
                            max = index[j];
                            pointer = j;
                        }
                    }
                }
                buffer[pointer] = references[i];
                fault++;
                pointer++;
                if (pointer == frames)
                {
                    pointer = 0;
                    isFull = true;
                }
            }
            for (int k = 0; k < frames; k++)
            {
                mem_layout[i][k] = buffer[k];
             }
        }

        System.out.println("The number of Hits: " + hit);
        System.out.println("Hit Ratio: " + (float) hit / ref_len);
        System.out.println("The number of faults: " + fault);
    }
}
```



```
Microsoft Windows [Version 10.0.19045.3208]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rashm\Desktop>javac FIFO.java

C:\Users\rashm\Desktop>java FIFO
Please enter the number of frames:
4
Please enter the length of the reference string:
10
Please enter the reference string:
7
0
1
2
0
3
0
4
2
3

  7   7   7   7   7   3   3   3   3   3
 -1   0   0   0   0   0   0   4   4   4
 -1  -1   1   1   1   1   1   1   1   1
 -1  -1  -1   2   2   2   2   2   2   2
The number of Hits: 4
Hit Ratio: 0.4
The number of Faults: 6
```



```
Microsoft Windows [Version 10.0.19045.3208]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rashm\Desktop>javac LRU.java

C:\Users\rashm\Desktop>java LRU
Please enter the number of frames:
4
Please enter the length of the reference string:
10
Please enter the reference string:
7
0
1
2
0
3
0
4
2
3

  7   7   7   7   7   3   3   3   3   3
 -1   0   0   0   0   0   0   0   0   0
 -1  -1   1   1   1   1   1   4   4   4
 -1  -1  -1   2   2   2   2   2   2   2
The number of Hits: 4
Hit Ratio: 0.4
The number of faults: 6
```



```
Microsoft Windows [Version 10.0.19045.3208]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rashm\Desktop>javac Optimal.java

C:\Users\rashm\Desktop>java Optimal
Please enter the number of frames:
4
Please enter the length of the References string:
10
Please enter the references string:
7
0
1
2
0
3
0
4
2
3
The number of Hits: 4
Hit Ratio: 0.4
The number of faults: 6
```